Amendments to the Claims:

Please amend the following claims as indicated below. Added text has been underlined, deleted text has been struck through.

1. (currently amended) A method of operating a multiplication circuit in a cryptographic application to compute, with fewer memory accesses, the product of two operands (X and Y) at least one of which is substantially wider than the multiplication circuit, the multiplication circuit having a pair of word-wide operand inputs and a two-word-wide product output, where a word is a specified number of bits, each of the operands composed of a plurality of contiguous ordered word-wide operand segments (x_k and y_m) characterized by specific weights (k and m), the multiplication circuit having access to a memory, the method comprising the steps of:

loading word-wide operand segments of the two operands in a specified order from the memory into the multiplication circuit, the multiplication circuit including at least two registers (RX and RY) having access to said memory for temporarily holding the loaded segments;

multiplying the loaded segments to obtain two-word-wide intermediate products, the intermediate products having a weight equal to the sum of weights of the loaded segments;

adding intermediate products of the same weight in an accumulator, the accumulator having a size of three words plus a number of carry bits sufficient for handling a specified maximum operand size, the accumulator connected to a two-word input register (RZ) for temporarily holding any previously added products of a specified weight and a two-word output register (RR) for holding results of an addition step, said registers (RZ and RR) having access to said memory; and

storing accumulated results from said output register (RR) back into said memory at least after accumulating all intermediate products of the specified weight;

wherein the specified order for loading operand segments into said registers is a sequence defined by the resulting intermediate product weights, wherein the multiplying step is done in successive groups of two adjacent product weights, with the sequence within a group being selected such that, other than a first multiply operation in a given group, at most one of the operand segments need be read from memory into one register (RX or RY), the other operand segment already being stored in the other register (RY or RX) from the immediately preceding multiply operation.

2. (original) The method of claim 1, wherein the accumulator input register (RZ) is loaded with a segment of an accumulate word (Z), where the segment is of the same weight as the intermediate products to be formed in said multiply step, whereby the method of operating executes a multiply-accumulate operation on a pair of multiply operands and an accumulate operand.

3. (original) The method of claim 1, wherein the sequence proceeds within a group in a zigzag pattern of steadily increasing operand segment weights.

4. (original) The method of claim 1, wherein the sequence proceeds within a group in a zigzag pattern of steadily decreasing operand segment weights.

5. (original) The method of claim 1, wherein the multiplication circuit further contains a set of internal cache registers, said cache registers being loaded from memory with operand segments that are frequently used in a multiply operation.

6. (original) The method of claim 5, wherein the sequence proceeds within a group beginning with a multiply operation with at least one operand segment stored in a cache register in a first zigzag direction of steadily increasing or decreasing operand segment weights, then jumps to a multiply operation of the group not yet performed that also has at least one operand segment stored in a cache register and proceeds in a second zigzag direction of steadily decreasing or increasing operand segment weights until all multiply operations for that group are completed.

7. (original) The method of claim 1 wherein the specified order is preprogrammed in firmware within an operations sequencer of the multiplication circuit, with operand word lengths being included as an input parameter for a multiply command.

8. (original) A multiplication circuit, comprising:

a multiply-accumulate (MAC) unit including a multiplier array with inputs receiving single-word operand segments to be multiplied to form a two-word intermediate product, and also including an accumulator circuit with a first two-word input for receiving intermediate products from the multiplier, a second two-word input for receiving an accumulate value, and an output of three words plus a number of carry bits sufficient for handling a specified maximum operand size for providing a sum of input values from the two-word inputs, the accumulator output also feeding two words back to the second two-word input;

a set of internal address registers for addressing a random-access memory;

a set of internal data registers (RX, RY, RZ, RR) connected to receive and transmit segments of said operands from and to said memory and also connected to said multiplier array and to said accumulator so as to supply operand segments to the inputs thereof and receive outputs therefrom, each of said operand segments having a specified weight; and

an operations sequencer for controlling the accessing of memory by said internal address and data registers and controlling the sequence of multiply and accumulate operations by said respective multiplier array and accumulator,

wherein said sequence is defined by the resulting product weights equal to the sum of the operand segment weights being multiplied, wherein the multiply operations are done in successive groups of two adjacent product weights, with the sequence within a group being selected such that, other than a first multiply operation in a given group, at most one of the operand segments need be read from memory into one register (RX or RY), the other operand segment already being stored in the other register (RY or RX) from the immediately preceding multiply operation.